GENERATING ANNUAL CAP88 INPUT FILES FOR LANL MONITORED STACKS

Purpose

This Air Quality Group procedure describes how data can be electronically transferred from the Air Quality group's MS-ACCESS database into CAP88 input files.

Scope

This procedure applies to the generation of electronic files to be used in calculating doses for Rad-NESHAP reporting purposes and for the annual report. This procedure is for use when input files are not hand-entered.

In this procedure

This procedure addresses the following major topics:

Hazard Control Plan

The hazard evaluation associated with this work is documented in HCP-ESH-17-Office Work.

Signatures

Prepared by: Keith W. Jacobson, ESH-17	Date: 5/23/01
Approved by: Dave Fuehne, Rad-NESHAP Project Leader	Date: 5/23/01
Approved by: Terry Morgan, QA Officer	Date: 5/24/01
Work authorized by: Doug Stavert, ESH-17 Group Leader	Date: 5/29/01

02/02/04

General information about this procedure

Attachments

This procedure has the following attachments:

Number	Attachment Title	No. of pages
1	Example Of Inp88.Sh Script	1
2	Fortran Program mak88.f	4
3	Example of cat88.sh script	1

History of revision

This table lists the revision history and effective dates of this procedure.

Revision	Date	Description of Changes
0	9/30/99	New document.
1	6/13/01	Changes to reflect new operating system.

Who requires training to this procedure?

The following personnel require training before implementing this procedure:

- document preparer

Annual retraining is required and will be by self-study (“reading”) training.

Training method

The training method for this procedure is **self-study** (“reading”) and is documented in accordance with the procedure for training (ESH-17-024).

General information, continued

Definitions specific to this procedure

K-shell script: A computer macro written for the Unix operating system using commands common to the K-shell (DOS-like) environment.

Source term: A phrase to describe the number of radionuclides, type of radionuclides and the quantity of radioactivity, in Ci, that is released from a stack.

References

The following documents are referenced in this procedure:

- ESH-17-024, “Personnel Training”
- ESH-17-501, “Dose Assessment Using CAP88”
- ESH-17-507, “Preparation of the Annual RAD-NESHAP Report”

Note

Actions specified within this procedure, unless preceded with “should” or “may,” are to be considered mandatory guidance (i.e., “shall”).

Transferring Stack Monitoring Data

Background

Generating CAP88 input files for dose modeling is an important task for the annual Rad-NESHAP compliance assessment. Increasingly, manipulation of data is being done solely by software. As discussed on page 12 in ESH-17-507, "Preparation of the Annual RAD-NESHAP Report", it is important to be able to document and verify that software applications function as intended. The following steps outline how data is electronically transferred and reformatted from the MS Access database(s) into the CAP88 software package. This procedure shows how to perform the steps necessary and serves to document the software used in the process.

Databases and files

Currently, data from three individual MS-ACCESS databases are required for input to CAP88: RADAIR, Flow Measurement , and Release_Points. Output from each of these is placed into intermediate files that will be read by a Fortran program to generate the input file for each stack. There are three main steps: run an MS Access query to obtain the data of interest, save the data to a file on the ESH-17 computer 'sibyl', and build input files using scripts and programs. "Sibyl" is an HP workstation maintained by the meteorology project in ESH-17.

The steps below assume the user has access to the all the databases and an account on 'sibyl' and has both applications "running".

Steps to create input files

To transfer data and create CAP88 input files, perform the following steps:

Step	Action
1	With regards to the radionuclide emissions, the data steward for the Annual Source Term on the RADAIR database releases an official memo and sends an electronic file of the calendar year emissions by release point and radionuclide, to the individual performing this procedure.
2	After the electronic file has been received and opened, use the Edit/Select all command and hit the Copy button on the menu bar.
3	Switch to the window for the 'sibyl' computer, and invoke an editor such as fred or VI. With the appropriate insert command and paste sequence, copy the data from the MS Windows clipboard to the editor. Save and close the file and name it "stk88.dat." Ensure the file is in the proper users directory (e.g., /users/kwj/cy00).

Steps continued on next page.

Transferring Stack Monitoring Data, continued

Step	Action
4	The file is now ready to be used by inp88.sh script and the mak88.f Fortran program that builds cap88 input files ('prepnpt') for each monitored stack.
5	With regards to the stack flow measurements, the data steward for the for the Flow Measurements database releases an official memo and sends an electronic file of the stack flows by release point to the individual performing this procedure.
6	After the electronic file has been received and opened, use the Edit/Select all command and hit the Copy button on the menu bar.
7	Switch to the window for the mainframe and invoke an editor such as fred or VI. With the appropriate insert command and paste sequence, copy the data from the windows clipboard to the editor. Save and close the file. Name it "flw98.dat" and ensure the file is in the proper directory (e.g., /users/kwj/cy00).
8	The file is now ready to be used by inp88.sh script and the mak88.f Fortran program that builds cap88 input files ('prepnpt') for each monitored stack.
9	Obtain access to the Release_Points database and identify the appropriate query(s) needed to extract the annual air emissions from the database. For the 2000 assessment year the query used was: SELECT DISTINCTROW stacks.ESIDNUM, stacks.Height, stacks.Diameter FROM stacks WHERE (((stacks.Monitored)=Yes))
10	After the query has been run, use the Edit/Select all command and hit the Copy button on the menu bar.
11	Switch to the window for the 'sibyl' and invoke an editor such as fred or VI. With the appropriate insert command and paste sequence, copy the data from the windows clipboard to the editor. Save and close the file. It should be named "phy88.dat". Ensure the file is in proper directory (e.g., /users/kwj/cy00).
12	The file is now ready to be used by inp88.sh script and the mak88.f Fortran program that builds cap88 input files ('prepnpt') for each monitored stack.
13	Obtain a copy of the K-shell script inp88.sh (Attachment 1) and run the script to pre-format the downloaded data for the mak88.f program (Attachment 2). Compile and run the mak88.f program. A set of intermediate prepnpt files is created.

Steps continued on next page.

Transferring Stack Monitoring Data, continued

Step	Action
14	Obtain a copy of the K-shell script cat88.sh (Attachment 3) and run the script to complete building of the prepnpt files for the CAP88 runs. These should be stored in a directory (e.g., /users/kwj/cy00) as well. At this point, the user is ready to begin making CAP88 runs for the annual report according to procedure ESH-17-501.

Modification of tritium source term

The user should be aware that the inp88.sh script performs a modification of the tritium source term. While the RADAIR database records both the HTO and gas form of tritium, CAP88 does not have a dose factor for the gas form. Thus, tritium activity reported as gas should be modeled as HTO. The inp88.sh script performs this step automatically.

Non-CAP88 radionuclides

At this time, the mak88.f program does not separate the non-CAP88 radionuclides from the source term into a separate file. The user must perform this step manually.

Records resulting from this procedure

Records

The following electronic records are generated as a result of this procedure.

- a CAP88 input file for each source term

These files are stored in an appropriate directory on the ‘Sibyl’ computer. Once the CAP88 output file has been printed and all reviews and signatures have been obtained for a particular source term, the files may be deleted.

Click here to record “self-study” training to this procedure.

EXAMPLE OF INP88.SH SCRIPT

inp88.sh K-shell script

```
#!/bin/ksh
# inp88.sh
#
# k-shell script to pre-format MS-ACCESS output for MAK88.F program
# 26-Apr-2001=last update by KwJ
#
rm stk88.txt flw88.txt phy88.txt
echo 'NOTE, first line (header), of each source file will be DELETED!'
cp /users/kwj/cap88/2000/stk88.dat stk88.txt
cp /users/kwj/cap88/2000/flw88.dat flw88.txt
cp /users/kwj/cap88/2000/phy88.dat phy88.txt
#
#reformat radionuclide source term file
tr '[:upper:]' '[:lower:]' < stk88.txt > tmp1
expand -1,10,20,30 < tmp1 > tmp2
#
sed '1d' < tmp2 > tmp3
sed 's/(hto)/     /g' < tmp3 > tmp4
sed 's/(gas)/     /g' < tmp4 > stk88.inp
rm tmp*
#
sort -n +1 -8 -o stk88.inp stk88.inp
#
awk '/ / { print $1 }' stk88.inp | uniq -d | wc -l > rev88.inp
awk '/ / { print $1 }' stk88.inp | uniq -c >> rev88.inp
#
#reformat stack parameters file
tr '[:upper:]' '[:lower:]' < phy88.txt > tmp1
expand -1,10,20 < tmp1 > tmp2
#
sed '1d' < tmp2 > phy88.inp
rm tmp*
#
#reformat exit velocity file
tr '[:upper:]' '[:lower:]' < flw88.txt > tmp1
expand -1,10,20 < tmp1 > tmp2
#
sed '1d' < tmp2 > flw88.inp
rm tmp*
#
echo 'End of inp88.sh, see => stk88.inp, phy88.inp, and flw88.inp'
```

FORTRAN PROGRAM MAK88.F

mak88.f Fortran Program

```
program mak88
***** Generates prepnpt input data for CAP88 program. ***********
***** 02-May-2000=last modified by KwJ, created April 1998 ******
***** ansi fortran 77, HP f77 compiler, sibyl-unix-os ***********
*
***** change names of input files (36-38) for respective year ***
***** stk??.inp = radionuclide input file                     ***
***** phy??.inp = stack height, diameter input file           ***
***** flw??.inp = stack flow rate input file                  ***
*
*
      integer arsz1, arsz2
      parameter ( arsz1 = 1000 , arsz2 = 100 )
      integer nrecs, no_radi(72), nopts, tot1, tot2, no_phys
      real ci(arsz1)
      character anlysis(arsz1)*7, esidnum(arsz1)*8
      character relpt(arsz2)*8
      character pid(arsz2)*8,ph(arsz2)*5,dia(arsz2)*5
      character fid(arsz2)*8,vel(arsz2)*5
      character ayear*4
      character frmt44*31,frmt55*21,frmt66*14
      common/rev_arr/relpt,no_radi
      common/inp_arr/anlysis,ci
      common/phy_arr/pid,ph,dia,fid,vel
c**** nopts: is the number of release points to generate file for
c**** no_radi: is the number of radionucs for each release point
c**** relpt(array): is the array of release point names
#     call getdate(date_and_time)
c**** frmt44 = format for emissions input file ****
c**************************************************
      frmt44='(t1,a8,t11,a7,t21,g10.2,t31,a4)'
      frmt55='(t1,a8,t11,a5,t21,a5)'
      frmt66='(t1,a8,t11,a5)'
c**************************************************

      open(unit=44,file='stk99.inp',status='unknown')
      open(unit=55,file='phy99.inp',status='unknown')
      open(unit=66,file='flw99.inp',status='unknown')

c**************************************************
      do 18  i = 1, arsz1
        read(44, frmt44, end=55, iostat=ios44, err=49)
     ,    esidnum(i), anlysis(i), ci(i), ayear
18    continue
c*************************************************
49    if (ios44 .eq. 3023 ) then
c       last line of data file is blank, ignore the error
      end if
      print*, ' iostat= ',ios44
55    nrecs = i - 1
      print*, 'The number of records in the file is...', nrecs
      print*, 'The first ESIDNUM is...', esidnum(1)
      print'(t1,a,t20,1pe10.3)', ' The Ci(1).......', Ci(1)
      print'(t1,a,t20,1pe10.3)', ' The Ci(nrecs)...', Ci(nrecs)
      call review_pts(nopts)
      print*, 'Number of release points counted were...',nopts
c*************************************************
      do 28  i=1, arsz2
        read(55,frmt55, end=135) pid(i),ph(i),dia(i)
28    continue
135   tot1=i-1
      do 38  i = 1, arsz2
        read(66,frmt66, end=140) fid(i), vel(i)
38    continue
140   tot2=i-1
      if( tot2 .gt. tot1 ) tot1=tot2
      no_phys = tot1
c*************************************************
      call build88(nopts, no_phys)
      end
c***** end of main program ***********************
c*************************************************
      subroutine review_pts(nopts)
      integer no_radi(72), nopts
      character relpt(100)*8, frmt22*13
      common/rev_arr/relpt,no_radi
      frmt22 = '(t3,i2,t6,a8)'
      open(22, file='rev88.inp', status='old')
      read(22, '(t1,i2)' ) nopts
      do 28  i = 1, nopts
        read(22, frmt22) no_radi(i), relpt(i)
        print*,
     ,  'Release point=',relpt(i),' no. radionuclides...',no_radi(i)
28    continue
      end
c*************************************************
      subroutine build88(nopts, no_phys)
c**** constructs CAP88 prepnpt info **************
c**** 30-Apr-1998 last modified ******************
      integer arsz1, arsz2, triml, nopts, no_phys
      parameter ( arsz1 = 1000 , arsz2 = 100 )
      integer no_radi(72), unit_no, inc1
c     integer inc2
      real ci(arsz1)
      character anlysis(arsz1)*7, id*8, nnuc*9
      character relpt(100)*8, phyinfo*39, date_and_time*36
      character pid(arsz2)*8,ph(arsz2)*5,dia(arsz2)*5
      character fid(arsz2)*8,vel(arsz2)*5
      common/rev_arr/relpt,no_radi
      common/inp_arr/anlysis,ci
      common/phy_arr/pid,ph,dia,fid,vel
      call getdate(date_and_time)
      inc1= 1
c     inc2= 0
      unit_no = 8800
      do 18  i = 1, nopts
c       inc2 = inc1 + no_radi(i)
        unit_no = unit_no + i
        id(1:8)=relpt(i)(1:8)
        call fname( id, unit_no )
        call stackit( id, no_phys, phyinfo )
        write(unit=unit_no, fmt='(a20)') 'physical source data'
        write(unit=unit_no, fmt='(a1)' ) '1'
        write(unit=unit_no, fmt='(a39)') phyinfo
        write(unit=unit_no, fmt='(a19)') 'wind frequency data'
        write(unit=unit_no, fmt='(a17)') 'radionuclide data'
        write(unit=unit_no, fmt='(i2)' ) no_radi(i)
        do 28  j = inc1, (inc1 + no_radi(i) - 1)
          nnuc=(anlysis(j)(1:triml(anlysis(j)))) // ''','
          write(unit=unit_no, fmt=222)
     ,    ' $radi nuc=''', nnuc, 'rel=', ci(j), ' $end'
28      continue
        inc1 = (inc1 + no_radi(i) )
        write(unit=unit_no, fmt='(a16)') 'population array'
        write(unit=unit_no, fmt='(a7)' ) 'comment'
        write(unit=unit_no, fmt='(a24)') '  input created by mak88'
        write(unit=unit_no,fmt='(a13,a36)')
     &   '  created on ', date_and_time
18    continue
222   format(t1,3(a),bn1pe10.2,a)
      end
c*************************************************
      function triml(string)
      character string*(*)
      integer triml, l
      logical nonblk
      nonblk = .false.
      do 18  l = len(string), 1, -1
        if( string(l:l) .ne. ' ' ) then
        nonblk = .true.
        goto 19
      end if
18    continue
19    if(nonblk)then
        triml=l
      else
        triml=0
      end if
      end
c*************************************************
      subroutine getdate (date_and_time)
      integer*4 today(3), now(3)
      character *8 ctime
      character*10 cdate
      character *36 date_and_time
      call idate(today)   ! today(1)=day, (2)=month, (3)=year
      call itime(now)     ! now(1)=hour, (2)=minute, (3)=second
c  convert integer to character string
      write (cdate,'(i2.2, "/", i2.2, "/", i4.4,)') today(2),today(1),
     &  today(3)
       write (ctime,'(i2.2, ":", i2.2, ":", i2.2,)') now
       date_and_time=cdate//' '//ctime
       return
       end
c*************************************************
      subroutine fname( id, unit_no)
      integer unit_no
      character filen*11, id*(*)
      filen(1:3)='pre'
      do 18  j = 1, 8
        filen(j+3:j+3) = id(j:j)
18      continue
      open(unit=unit_no,file=filen,status='unknown')
      end
c*************************************************
      subroutine stackit( id, no_phys, phyinfo)
c****  merging stack data with source term *****
c****  writing formated stack data line    *****
      integer arsz1, arsz2, no_phys
      parameter ( arsz1 = 1000 , arsz2 = 100 )
      character phyinfo*39, id*8
      character pid(arsz2)*8,ph(arsz2)*5,dia(arsz2)*5
      character fid(arsz2)*8,vel(arsz2)*5
      common/phy_arr/pid,ph,dia,fid,vel
c     print*, ' no_phys = ', no_phys, ' id= ', id
      do 38  i=1, no_phys
        if( pid(i) .eq. id ) then
          write(phyinfo, fmt=300)
     ,    ' $phys ph=',ph(i),'dia=',dia(i),'vel=',vel(i),' $end'
        else
c         don't do anything
        end if
38    continue
300   format(t1,a10,a5,x,a4,a5,a4,a5,a5)
      end
c**** end of program mak88.f *********************
```

EXAMPLE OF CAT88.SH SCRIPT

cat88.sh K-shell script

```
#!/bin/ksh#!/bin/ksh
# cat88.sh
#
# shell script to merge/format individual prepnpt files.
#
cp /users/kwj/cap88/2000/pre-head pre-head
echo "Input the name of the input file, eg. pre..."
#set fname = $<
read fname
cat pre-head $fname > prepnpt
cp prepnpt $fname
echo '**** end of cat88.sh script, printing file on screen ****'
cat $fname

# cat88.sh
#
# shell script to merge/format individual prepnpt files.
#
cp /users/kwj/cap88/2000/pre-head pre-head
echo "Input the name of the input file, eg. pre..."
#set fname = $<
read fname
cat pre-head $fname > prepnpt
cp prepnpt $fname
echo '**** end of cat88.sh script, printing file on screen ****'
cat $fname
```